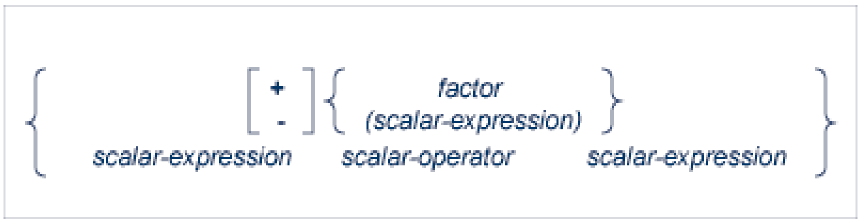
Scalar Expressions

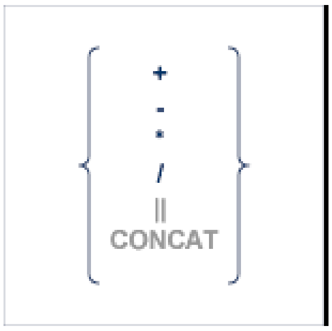
A *scalar-expression* consists of a factor or other scalar expressions including scalar operators.



Concerning reference priority, scalar expressions behave as follows: When a non-qualified variable name is specified in a scalar expression, the first approach is to resolve the variable name as column name of the referenced table. If no column with the specified name is available in the referenced table, Natural tries to resolve this variable as a Natural user-defined variable (host variable).

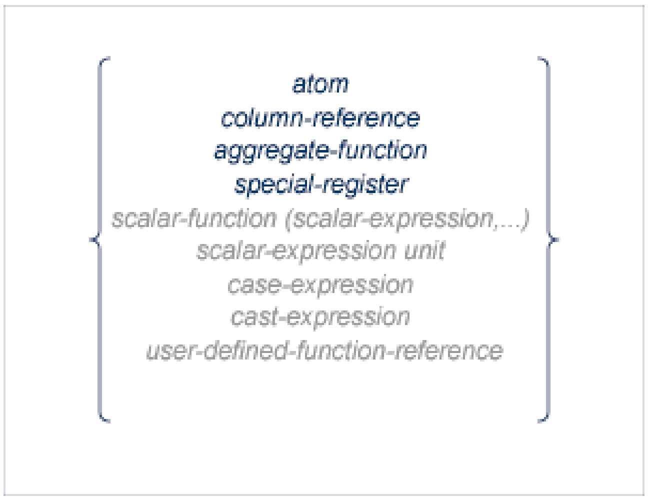
---

scalar-operator



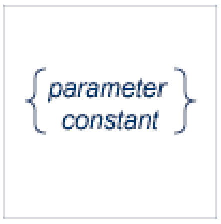
A *scalar-operator* can be any of the operators listed above; the operators "-" and "/" must be separated by at least one blank from preceding operators.

factor



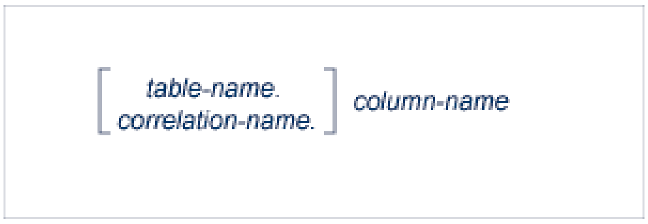
A *factor* can consist of one of the items listed in the above diagram.

atom



An *atom* can be either a *parameter* or a *constant*; see also the section Basic Syntactical Items.

column-reference

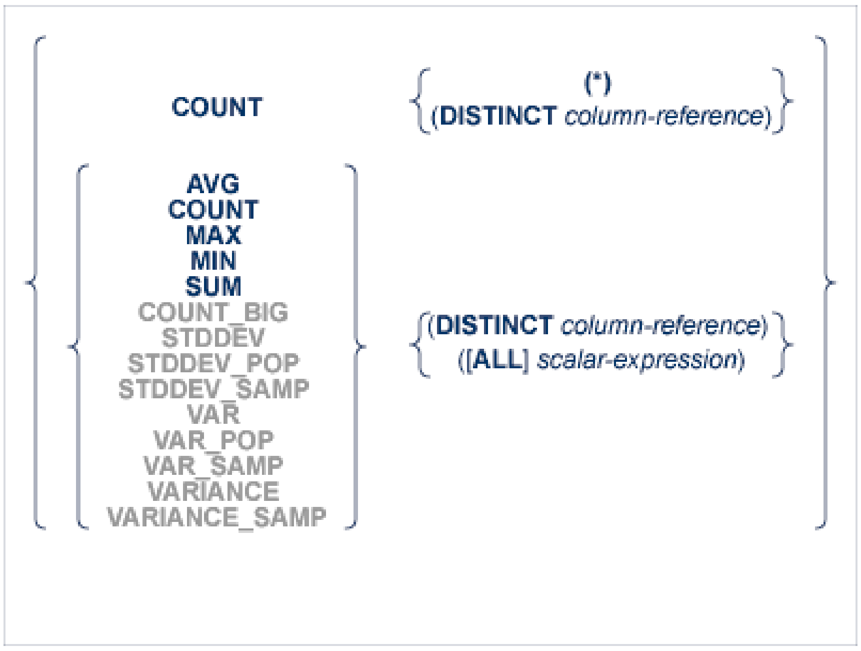


A column-reference is a column name optionally qualified by either a table-name or a correlation-name (see also the section Basic Syntactical Items). Qualified names are often clearer than unqualified names and sometimes they are essential.

Note:
A table name in this context must not be qualified explicitly with an authorization identifier. Use a correlation name instead if you need a qualified table name.

If a column is referenced by a *table-name* or *correlation-name*, it must be contained in the corresponding table. If neither a *table-name* nor a *correlation-name* is specified, the respective column must be in one of the tables specified in the FROM clause.

aggregate-function



SQL provides a number of special functions to enhance its basic retrieval power. The so-called SQL aggregate functions currently available and supported by Natural are:

- **AVG** which gives the average of the values in a column,
- **COUNT** which gives the number of values in a column,
- **MAX** which gives the highest value in a column,
- **MIN** which gives the lowest value in a column,
- **SUM** which gives the sum of the values in a column.

Apart from COUNT(*), each of these functions operates on the collection of scalar values in an argument (that is, a single column or a *scalar-expression*) and produces a scalar value as its result.

Example:

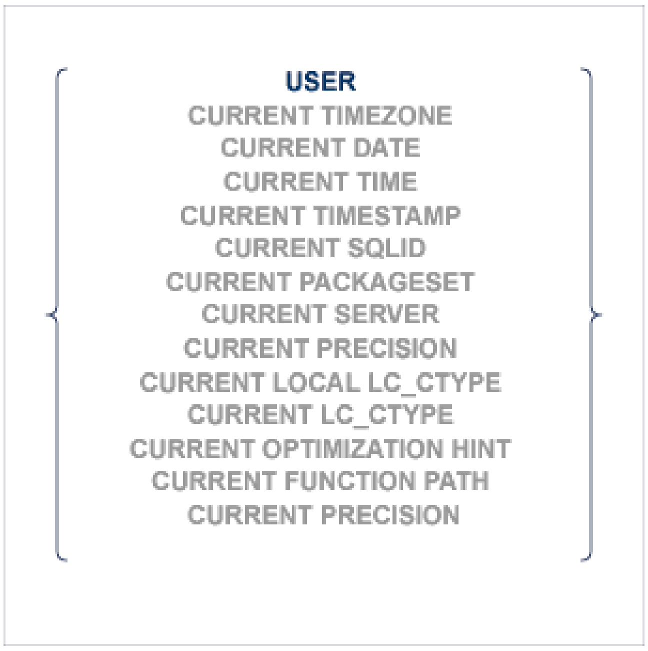
```
DEFINE DATA LOCAL
1 AVGAGE   (I2)
END-DEFINE
...
SELECT AVG (AGE)
  INTO AVGAGE
  FROM SQL-PERSONNEL
  ...
```

In general, the argument can optionally be preceded by the keyword DISTINCT to eliminate redundant duplicate values before the function is applied.

If DISTINCT is specified, the argument must be the name of a single column; if DISTINCT is omitted, the argument can consist of a general *scalar-expression*.

DISTINCT is not allowed with the special function COUNT(*), which is provided to count all rows without eliminating any duplicates.

special-register



```
USER
CURRENT TIMEZONE
CURRENT DATE
CURRENT TIME
CURRENT TIMESTAMP
CURRENT SQLID
CURRENT PACKAGESET
CURRENT SERVER
CURRENT PRECISION
CURRENT LOCAL LC_CTYPE
CURRENT LC_CTYPE
CURRENT OPTIMIZATION HINT
CURRENT FUNCTION PATH
CURRENT PRECISION
```

A reference to a *special-register* returns a scalar value.

With the exception of USER, *special-registers* do not conform to standard SQL and are therefore supported by the Natural SQL Extended Set only.

scalar-function

USER
ABS
ABSVAL
ACOS
ADD_MONTHS
ASIN
ATAN
ATAN2
ATANH
BLOB
CCSID_ENCODING
CEIL
CEILING
CHAR
CLOB
COALESCE
CONCAT
COS
COSH
DATE
DAY
DAYOFMONTH
DAYOFWEEK
DAYOFWEEK_ISO
DAYOFYEAR
DAYS
DBCLOB
DEC
DECIMAL
DEGREES
DIGITS
DOUBLE

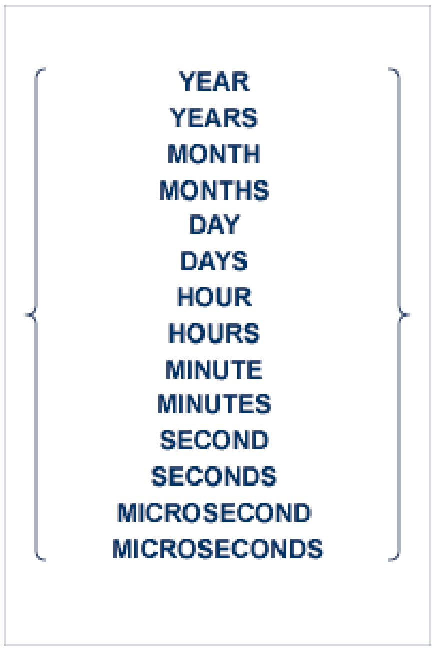
DOUBLE_PRECISION
EXP
FLOAT
FLOOR
GRAPHIC
HEX
HOUR
IDENTITY_VAL_LOCAL
IFNULL
INSERT
INTEGER
JULIAN_DAY
LAST_DAY
LCASE
LEFT
LENGTH
LN
LOCATE
LOG
LOG10
LOWER
LTRIM
MAX
MICROSECOND
MIDNIGHT_SECONDS
MIN
MINUTE
MOD
MONTH
MULTIPLY_ALT
NEXT_DAY
NULLIF

POSSTR
POWER
QUARTER
RADIANS
RAISE_ERROR
RAND
REAL
REPEAT
REPLACE
RIGHT
ROUND
ROUND_TIMESTAMP
ROWID
RTRIM
SECOND
SIGN
SIN
SINH
SMALLINT
SPACE
SQRT
STRIP
SUBSTR
TAN
TANH
TIME
TIMESTAMP
TIMESTAMP_FORMAT
TO_CHAR
TO_DATE
TRANSLATE
TRUNC

TRUNC_TIMESTAMP
TRUNCATE
UCASE
UPPER
VALUE
VARCHAR
VARCHAR FORMAT
VARGRAPHIC
WEEK
WEEK_ISO
YEAR

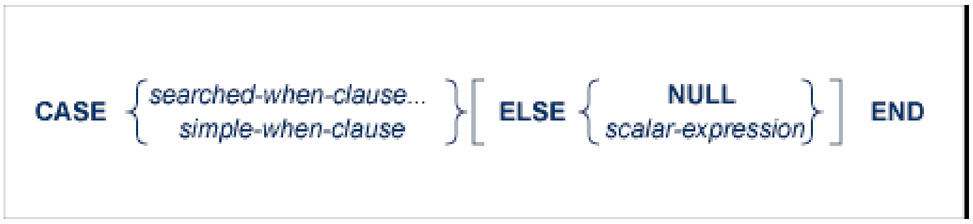
A *scalar-function* is a built-in function that can be used in the construction of scalar computational expressions. The above *scalar-functions* are supported by the Natural SQL Extended Set.

units



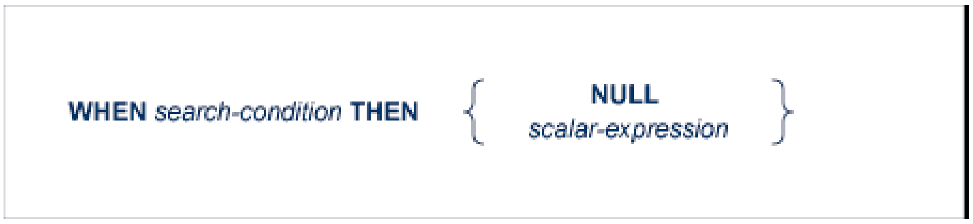
Units do not conform to standard SQL and are therefore supported by the Natural SQL Extended Set only.

case-expression



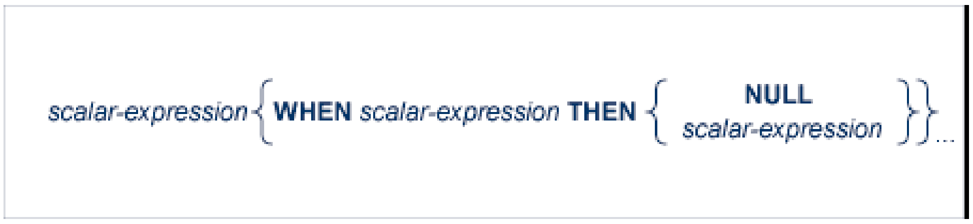
case-expressions do not conform to standard SQL and are therefore supported by the Natural SQL Extended Set only.

searched-when-clause



See details on *search-condition*.

simple-when-clause



cast-expression

```
CAST ( scalar-expression AS data-type )
```

cast-expressions do not conform to standard SQL and are therefore only supported by the Natural SQL Extended Set.

user-defined-function-reference

The option *user-defined-function-reference* belongs to the Natural SQL Extended Set. This options allows you to invoke any user-defined function. Arguments have to be placed inbrackets and separated by commas. The user-defined function must be declared in the target RDBMS.